



Agile/Scrum Implemented in Large-Scale Distributed Program

Executive Summary

It was early July 2010 when problems were detected while running a large program at one of our clients in the insurance industry. The program appeared to be going in circles and was conspicuously stuck in requirements and analyses. It was not until late July when it was decided to move the program to an Agile/Scrum approach; the plan was to get the program back on track for a delivery in May 2012. As expected, numerous questions were raised. Among them: How long should the Sprints be? How do we put a backlog together quickly? How do we get the teams to be productive as soon as possible? Do we include testing within the Sprint teams? How do we finish the analyses of the outstanding requirements? Etc., etc., etc.

Time was of the essence. Therefore, the immediate focus was placed on productivity rather than trying to answer all questions up front – which would have not been feasible in the first place. Nevertheless, the leadership team in the next two weeks tried to answer as many of the outstanding questions as possible before the first Sprint kicked off. This white paper describes how we met the challenge.

The Two-Week Planning

Upon further discussion, we decided to have two distributed teams between offshore and on-site. The Sprints would be four weeks in duration. The

teams decided to go with four-week sprints as we still had some open requirement clarifications which would need to be solved in the Sprint or moved out to the support team. The goal was to resolve all clarifications within the Sprint teams, as speed was required.

Original Sprint Teams

On-site	Offshore
Proxy Sprint Lead	Sprint Lead Lead/ Scrum Master
Business Analyst	Business Analyst
Lead Tester	Tester
UI Architect	Service Architect
Not Applicable	Five Developers

Figure 1

During the first retrospective, it became apparent that having the on-site support resources tied directly to the Sprint team was not working. These resources were moving across both teams, the idea being to help remove blockers and issues that the teams were running into. But since this was not working these support resources, including the offshore support resources, were moved into a new team called the support team. This team spent 50% of its time working for the current Sprint and 50% working on the outstand-

ing analyses items present at the beginning. This proved to be a successful move.

Support Team

On-site	Offshore
Sprint Lead/ Scrum Master	Not Applicable
Business Analyst	Business Analyst
UI Architect	Service Architect
Service Architect	UI Architect

Figure 2

Efforts were made to cut down on the idle time to gain efficiency. To achieve this, business analysts and architects offshore were put in place to speed up the program and were placed close to the Sprint teams performing the work. The goal was to resolve 75% to 90% of all team-raised issues before the issues came to be on-site. This approach produced better results and velocity with the Sprint teams.

Testing

Upon looking at the testing, the following issues were carefully reviewed:

- What would be required to complete the application and roll the application out to production?
- Could we develop user stories in sequence?
 - If we could complete the user stories in sequence, we could execute integration testing during or in the next Sprint.

Based on the answers to these issues, we decided that both the unit testing and feature testing would be part of the Sprints. System integration and user accepted testing would occur when the program is completed and will follow the client's current process for applications to go into production.

Product Backlog

The creation of the product backlog became an easy answer, or so we thought, as the program was already in progress and all the work items were broken out into features. We decided to make these features the work items or user stories for the Sprint teams.

The first Sprint kicked off in early August. At that moment the architects, program sponsor and the

delivery lead were taken from on-site to offshore for four weeks. There were three parts to the first Sprint; the two Sprint teams minus the architects worked on a very small feature set. This allowed the delivery lead (Agile/Scrum coach) and architects to train the Sprint teams. Furthermore, it allowed both the architects and business analysts to work the product backlog confirming the user stories were the right size to be completed in one Sprint and that all issues were logged for the support resources to work.

After the second Sprint, a support team was created and decided that it would need its own product backlog. Two reasons lead to that decision:

- The team's workload would be managed more efficiently and effectively.
- The Sprint team's velocity would not be impacted should the support team's tasks be created under the feature in the Sprint team's backlog.

Sprints

During the planning phase, it was decided to go with four-week Sprints, which allowed time to complete design, unit testing, user story test case design and test case execution. What was found in every retrospective for the first six Sprints was that the Sprints seemed to operate as mini waterfalls.

Testing would require everything to be perfect in the environment before executing the test cases. It took a little bit of work and training to get testing away from this approach. However, another issue arose. Once completed, development was not always releasing the feature/user story; they had begun to wait until everything was ready. Communication became critical to make things work more efficiently in the Sprint. This successfully ensured that developers were telling testing when a feature/user story was ready, thus guaranteeing the right focus on test case design.

Scoping

Two days were allocated at the beginning of each Sprint for the teams to decide what was in scope and the reviewing of the earlier estimates. In these two days, the teams reviewed the feature, ensuring there were no issues/blockers that would hold up the Sprint. Should any issues/blockers be found, the feature would be marked as blocked and a new feature would be placed on the support team's backlog to be worked. Next, the team would break out the feature into tasks

that needed to be completed during the Sprint and estimate these tasks accordingly. The feature then would be allocated to the team and the tasks would be added to rally for tracking.

Scrum of Scrum

During Sprints one and two, the team's stand-up meeting would include the program's leadership on the phone posing three questions to the team members:

- What did you work on yesterday?
- What will you work on today?
- Do you have any issues/blockers keeping you from completing your work?

Although it was great to hear all the members give their status, this process proved to be very ineffective. Furthermore, it was costing the team valuable time as the stand-up would go for more than 30 minutes every day. This approach was soon changed, after the second Sprint, allowing the team to hold their stand-up each day without leadership present. After the meeting, the Sprint lead/Scrum master would send out a status report outline and current status of the Sprint. Each day the leadership would have a Scrum of Scrum¹ meeting where the Sprint leads/Scrum masters would review the current status and raise any blockers their teams were having. At the end of this meeting, the on-site delivery lead would publish a combined status for the Sprint and action plans for the blockers that were raised in the Scrum of Scrum meeting.

Retrospective

At the beginning, we decided that all members from all teams would participate in one retrospective. At first, with only two teams, this worked fine. However, it was a challenge to complete this in an hour. And hearing from everyone was increasingly difficult as two more Sprint teams were added and the velocity was picking up. In the mean time, other issues were arising in the retrospective of higher importance (or at least they seemed to be of higher importance). Thus, the issue of hearing everyone did not get addressed until we moved into day 2 of the program. To make things more efficient and effective, the "retrospective of retrospective" was created; this is elaborated on in the next steps section.

Demo

Many eyes were on this program from the get-go. Consequently, we decided to have the lead business analyst perform the demos to the leadership team instead of the teams themselves.

No demo was done until we had completed four Sprints. This allowed us to get enough substance to the product before showing it to the executive team.

The program had such visibility at the executive level that requests for demos were pouring in almost daily. A line had to be drawn on these requests, as they were having a dramatic impact on the QA environment – especially given that the integration and production environments were not ready yet.

Next Steps

Planning for day 2 began in late April 2011, addressing some of the big issues that arose during day 1. This was the main objective of this phase. We decided to address three items at this time: scoping was taking too long for each Sprint; demos should be done by the developer who coded the feature/user story instead of the lead BA; and the retrospective was taking too much time and making it hard for us to hear from everyone.

Backlog Grooming

The implementation of the backlog grooming was set to fix the scoping issue. The first step was to get a product owner – or at least a proxy product owner – who could efficiently set business priorities. Although we were not successful getting a product owner, we did get a proxy product owner named.

To kick this off, we started having a grooming session prior to the completion of day 1 and the start of day 2. In these sessions both the support team and the product owner were directly reviewing and ensuring the backlog logs user stories were the right length and the Sprint teams could accordingly code from them. When any issue was identified, it was immediately sent to the business analysts to be fixed prior to the beginning of day 2. Moreover, we had the proxy product owner establish the initial business priority for the user stories and review it with the business analysts, thus making sure we knew what their priority was once day 2 began.

Furthermore, we decided that during the first couple of Sprints of day 2, twice-weekly meetings were going to be held with a limit of one hour per meeting.

Demos

Day 2 was much smaller than day 1: Five Sprints running four weeks in length compared with 15

Sprints for day 1. Therefore, the team demos for the proxy product owner were implemented and no demos for the leadership team were allowed until the fourth Sprint was completed. Now that we had implemented day 1 to production, the executive team and the leadership team were willing to step back some and wait for their demos.

Retrospective of Retrospective

A lot of wasted time was experienced in the retrospective meeting during day 1, as all team members involved were trying to give their input. This in turn made it impossible for everyone to be heard in the allotted time given for the meeting. To correct the problem, we decided to have the Sprint leads/Scrum master run their team retrospective. Notes were made on things that worked well and did not work.

Demos were held at 7:30 a.m. on the last day of the Sprint on-site and the retrospective of retrospective at 8:30 a.m. The retrospective meeting was changed to a similar format as the Scrum of Scrum. In this new format, the Sprint leads would represent their teams and only the leadership would be there. This new, effective format was called the Retrospective of Retrospective. It allowed us to focus on what truly worked and what did not. Furthermore, we came out from the meeting with an action plan for the top five items.

Daikibo Validated

Daikibo is our large-scale Agile/Scrum framework. Daikibo states that we should separate cross-functional teams and bifurcated responsibilities (a producer/consumer model) operating in an incremental/iterative pipeline approach,

following Agile principles with Scrum. The teams will be organized in three areas: concept team, delivery team and the system integration test team. The concept team is responsible for the story production/generation, the delivery team will consume the stories and the system integration test team will validate the stories.

When one compares the approach we implemented at this client to the Daikibo framework, three differences are clear. First, there is no integration Sprint following the first development Sprint. The reason for this, as noted above, is that features/user stories were not developed in sequence. Second, given how the process was handled during the requirements and analyses phase prior to the move to Agile/Scrum, we had no need for a formal Sprint 0. Third, we had a support team instead of a concept team. This team was distributed between on-site and offshore in an attempt to clear issues/blockers as quickly as possible. Daikibo is merely a framework, and therefore will not work for every program as outlined.

Conclusion

The program was delivered one sprint (or four weeks) later than originally planned. The second phase of the program has been now running for 12 weeks and it is currently ahead of schedule and expected to deliver early. This approach opened up communication at all levels; program sponsors knew weekly if we were on schedule or not. The teams were able to raise issues/blockers and know that they would be either removed quickly or the feature would be moved to another Sprint. If the program and organization can be agile with a little "a" then I would say the program will be successful.

Footnote

¹ <http://learnssoftwareprocesses.com/2010/03/02/scrum-of-scrums-a-brief-definition-and-some-details/>

About the Author

Dwayne Gifford is an Associate Director in Cognizant's Advanced Solutions Group and has 20-plus years of software development experience. He is an experienced Agile practitioner who has led some of Cognizant's largest distributed deliveries using Agile. He is working on his masters in computer science at Troy University. He can be reached at Dwayne.Gifford@cognizant.com.



About Cognizant

Cognizant (NASDAQ: CTSH) is a leading provider of information technology, consulting, and business process outsourcing services, dedicated to helping the world's leading companies build stronger businesses. Headquartered in Teaneck, New Jersey (U.S.), Cognizant combines a passion for client satisfaction, technology innovation, deep industry and business process expertise, and a global, collaborative workforce that embodies the future of work. With over 50 delivery centers worldwide and approximately 130,000 employees as of September 30, 2011, Cognizant is a member of the NASDAQ-100, the S&P 500, the Forbes Global 2000, and the Fortune 500 and is ranked among the top performing and fastest growing companies in the world. Visit us online at www.cognizant.com or follow us on Twitter: Cognizant.



World Headquarters

500 Frank W. Burr Blvd.
Teaneck, NJ 07666 USA
Phone: +1 201 801 0233
Fax: +1 201 801 0243
Toll Free: +1 888 937 3277
Email: inquiry@cognizant.com

European Headquarters

1 Kingdom Street
Paddington Central
London W2 6BD
Phone: +44 (0) 20 7297 7600
Fax: +44 (0) 20 7121 0102
Email: infouk@cognizant.com

India Operations Headquarters

#5/535, Old Mahabalipuram Road
Okkiyam Pettai, Thoraipakkam
Chennai, 600 096 India
Phone: +91 (0) 44 4209 6000
Fax: +91 (0) 44 4209 6060
Email: inquiryindia@cognizant.com